AMENDMENTS TO THE CLAIMS

Claims 4, 13, and 22, have been canceled without prejudice or disclaimer, and claims 1, 10, 19, and 20 have been amended as denoted in the following listing. Claims 23-26 have been added. This listing of claims will replace all prior versions and listings of claims in the application:

1. (Currently Amended) A method of processing data comprising:

defining an object with an option data structure which supports references to option values without preallocation of memory space for the full option values, wherein the object is instantiated from a class within a class inheritance hierarchy; and

notifying the object ~~objects of~~ a change in an option value of an option through change handlers ~~a change handler~~ identified by an option binding, the option binding being located by first searching a mapping data structure for a previously computed mapping to the option binding and, if no mapping was previously computed, by then computing the mapping to the option binding and storing the mapping in the mapping data structure,

wherein code for the change handlers for the option may be ~~is~~ defined in different classes within the class inheritance hierarchy ~~and the change handler code from each of the different classes is executed when the option value changes~~.

2. (Original) A method as claimed in claim 1 wherein the mapping data structure is a hash table.

3. (Original) A method as claimed in claim 1 wherein the option binding is a most specific option binding given a class and a base option binding.

4. (Canceled).

5. (Original) A method as claimed in claim 1 wherein an option data structure includes a default value, the method further comprising, in a get operation to an instance of the class, if an option value which applies to the instance has been set, getting the set option value and, if a value which applies has not been set, getting the default value for the class.

6. (Original) A method as claimed in claim 1 wherein the option data structure comprises a linked list of option items having option values.

7. (Original) A method as claimed in claim 1 wherein a nonlocal option value applies to other objects in a nonlocal option hierarchy.

8. (Original) A method as claimed in claim 7 wherein the nonlocal option hierarchy is a graphical hierarchy.

9. (Original) A method as claimed in claim 1 wherein the class which supports the option data structure includes defined fields to support values in preallocated memory space.

10. (Currently Amended) A data processing system comprising:

an object with an option data structure which supports references to option values without preallocation of memory space for the full option values, wherein the object is instantiated from a class within a class inheritance hierarchy;

~~a~~ change handlers which notify the object ~~notifies objects~~ of a change in an option value of an option;

an option binding which identifies one of said change handlers ~~a change handler~~; and

a mapping data structure which maps an option name and class to the option binding, wherein the option binding is located by first searching the mapping data structure for a previously computed mapping to the option binding and, if no mapping was previously computed, by then computing the mapping to the option binding and storing the mapping in the mapping data structure,

wherein code for the change handlers for the option may be ~~is~~ defined in different classes within the class inheritance hierarchy ~~and the change handler code from each of the different classes is executed when the option value changes~~.

11. (Original) A system as claimed in claim 10 wherein the mapping data structure is a hash table.

12. (Original) A system as claimed in claim 10 wherein the option binding is a most specific option binding given a class and a base option binding.

13. (Canceled).

14. (Original) A system as claimed in claim 10 wherein an option value data structure includes a default value which is obtained when an option value has not been set in an applicable instance object.

15. (Original) A system as claimed in claim 10 wherein the option data structure comprises a linked list of option items having option values.

16. (Original) A system as claimed in claim 10 wherein a nonlocal option value applies to other objects in a nonlocal option hierarchy.

17. (Original) A system as claimed in claim 16 wherein the nonlocal option hierarchy is a graphical hierarchy.

18. (Original) A system as claimed in claim 10 wherein the class which supports the option data structure includes defined fields to support values in preallocated memory space.

19. (Currently Amended) A data processing system comprising:

means for defining an object with an option data structure which supports references to option values without preallocation of memory space for the full option values, wherein the object is instantiated from a class within a class inheritance hierarchy; and

means for notifying the object ~~objects~~ of a change in an option value of an option through change handlers ~~a change handler~~ identified by an option binding, the option binding being located by first searching a mapping data structure for a previously computed mapping to the option binding and, if no mapping was previously computed, by then computing the mapping to the option binding and storing the mapping in the mapping data structure,

wherein code for the change handler for the option may be ~~is~~ defined in different classes within the class inheritance hierarchy ~~and the change handler code from each of the different classes is executed when the option value changes~~.

20. (Currently Amended) A computer program product comprising:

a computer usable medium for storing data; and

a set of computer program instructions embodied on the computer usable medium, including instructions to:

define an object with an option data structure which supports references to option values without preallocation of memory space for the full option values, wherein the object is instantiated from a class within a class inheritance hierarchy; and

notify the object ~~objects~~ of a change in an option value of an option through change handlers ~~a change handler~~ identified by an option binding, the option binding being located by first searching a mapping data structure for a previously computed mapping to the option binding and, if no mapping was previously computed, by then computing the mapping to the option binding and storing the mapping in the mapping data structure,

wherein code for the change handlers for the option is defined in different classes within the class inheritance hierarchy ~~and the change handler code from each of the different classes is executed when the option value changes~~.

21. (Original) A product as claimed in claim 20 wherein the option data structure comprises a linked list of option items having option values.

22. (Canceled).

23. (New) The method of claim 1, wherein the code for one or more of the change handlers is executed when the option value changes.

24. (New) The system of claim 10, wherein the code for one or more of the change handlers is executed when the option value changes.

25. (New) The system of claim 19, wherein the code for one or more of the change handlers is executed when the option value changes.

26. (New) The product of claim 20, wherein the code for one or more of the change handlers is executed when the option value changes.